完全データからのSimple Regular言語族の多項式時間反駁推論について(アルゴリズムと計算量理論)

AUTHOR(S):

渡辺, 紀仁; 佐藤, 優子

CITATION:

渡辺, 紀仁 ...[et al]. 完全データからのSimple Regular言語族の多項式時間反駁推論について(アルゴリズムと計算量理論). 数理解析研究所講究録 1995, 906: 228-235

ISSUE DATE:

1995-04

URL:

http://hdl.handle.net/2433/59439

数理解析研究所講究録
906巻 1995年 228-235

完全データからの Simple Regular 言語族の多項式時間反駁推論について

渡辺紀仁　　佐藤優子
Norihito WATANABE　　Masako SATO
大阪府立大学 総合科学研究科

Abstract

最近、Mukouchi & Arikawa により、機械発見の枠組みとして反駁推論が提案された。本稿では、Simple regular と呼ばれる正則言語の部分族が完全データから多項式時間反駁推論可能であることを示す。また、Simple regular 言語の特徴付け定理を証明し、この定理を用いて、Simple regular 言語族の Closure property 及び Angluin による reversible 言語との関係を与える。

1 はじめに

1967年、Gold[3] は「極限における同定」という枠組みで帰納推論の数学的モデルを提案し、形式言語と帰納的関数の帰納推論に理論的基盤を与えた。形式言語の帰納推論では、推論機械と呼ばれる実行的手続きが目標言語に関する例を次々と要求し、仮説或いは推測と呼ばれる、言語に対応する出力を次々と生成する。その仮説は言語に対応するものであり、例えば、文法やオートマトンなどがこれにあたる。推論機械が出力しうる仮説の集合を仮説空間と呼ぶ。仮説空間が帰納的言語の添字付き族のとき、仮説空間に属する任意の言語は完全データから極限において同定可能であることが示されている (cf. [3])。しかし、仮説空間に属さない言語を提示した場合、その推論機械は正解を出力することができない。

では、仮説空間に属さない言語の例を与えた場合、推論機械がどのように振る舞うのが望ましいだろうか。最近、Mukouchi & Arikawa [9] は、機械発見の枠組みとして、目標言語が仮説空間に存在しない場合、有限個の例から仮説空間を反駁 (refute) して停止することを要求する反駁推論を導入した。反駁推論可能となる言語族の特徴付け定理も得られており、この定理から Chomsky 階層の最下部に位置する正則言語族さえ、完全データから反駁推論可能でないことが示されている (cf.[9])。正則言語に関しては、*word* 上の正則表現で演算回数を高々 n 回に制限した正則言語の部分族は、完全データから反駁推論可能となることが言えている (cf. [6])。

一方、Mukouchi & Arikawa で示された反駁推論のアルゴリズムは、効率という観点からは現実的なものではなく、実際の機械発見に応用することはきわめて困難である。本稿では、効率よく反駁推論可能となる言語族について議論する。ここでは、反駁推論機械が例を受け取ってから、推測を出力するか、または、仮説空間を反駁するまでの計算時間が、それまでの入力の長さの多項式になるようなものを効率的であると考えることにする。これは、極限同定において一般的に受け入れられている多項式時間推論を反駁推論に適用したものであり、多項式時間反駁推論と呼ぶことにする。

本稿で扱う言語族は、K.Sato & M.Sato [4] により導入された *Simple regular* と呼ばれる正則言語 (以下、SR 言語) の族である。この族は各先頭文字が異なる *word* の出現回数が一回であるような正則表現で表される正則言語を含み、Tanida & Yokomori の Very regular 言語族 [7] を真に包

含する (cf. [4])。論文 [4] では、正データからの SR 言語の多項式時間極限同定可能性が示されている。本稿では、SR 言語族が完全データから多項式時間反駁推論可能であることを示す。

紙面の都合で定理・補助定理の証明は殆ど省略している。

2 準備

この節では、形式言語の完全データからの帰納推論に関する基本的な定義とこれまでに得られているいくつかの結果について述べる。

Σをアルファベットと呼ばれる空でない有限集合とし、特に断りのない限り $\{a_1, \cdots, a_m\}$ で表す。Σ上の有限な文字列を *word*、$w, u, v, w_1, u_1, v_1, \cdots$などで表す。$\Sigma$上の *word* の全体を$\Sigma^*$で表し、$\Sigma^*$から *empty word* と呼ばれる長さ 0 の語（λで表す）を除いたものをΣ^+で表す。Σ^*の部分集合を言語 (language) といい、L, L'等で表す。また、$N = \{ i \mid i \geq 1 \}$ とする。$w = uv$ $(u, v \in \Sigma^*)$ のとき、u を wの *prefix* と呼ぶ。

言語の族$\mathcal{L} = L_1, L_2, \cdots$が帰納的言語の添字付き族であるとは次のような帰納的関数 $f : N \times \Sigma^* \to \{0, 1\}$ が存在することをいう：

$$f(i, w) = \begin{cases} 1, & w \in L_i \text{のとき} \\ 0, & w \notin L_i \text{のとき}. \end{cases}$$

以降、帰納的言語の添字付き族を扱う。

$\Sigma^* \times \{0, 1\}$ の要素の無限列 $(w_1, t_1), (w_2, t_2), \cdots$が言語 L の完全提示であるとは、$L = \{ w_n \mid t_n = 1, n \geq 0 \}$ かつ $L^c = \{ w_n \mid t_n = 0, n \geq 0 \}$ となることをいう。

推論機械 (Inference Machine) とは、ときどき入力を要求してときどき正整数を出力する実行的な手続きである。推論機械の出力を仮説といい、推論機械が出力しうる仮説の集合を仮説空間と呼ぶ。

定義 2.1 [3] 推論機械 M が言語 L を完全データから極限同定するとは、L の任意の完全提示 σ に対して M の入力要求に応じてσ の例を次々と入力したときに M の生成する仮説の列 $j_1, j_2, \cdots$ が $L_j = L$ となる j に収束することをいう。

Arikawa & Mukouchi [9] によって導入された反駁推論機械（RIIM と略記）とは、ときどき入力を要求してときどき正整数を出力するか、または、仮説空間を反駁して停止する実行的手続きのことをいう。

定義 2.2 [9] RIIM M が σから言語族 $\mathcal{L}$（仮説空間）を反駁するとは、σの例を有限個入力した後に Mが停止することをいう。RIIM Mが完全データから言語族 $\mathcal{L}$ を反駁推論するとは、任意の言語 L と L の任意の完全提示σ に対して

(1) $L \in \mathcal{L}$ ならば、M は σから L を極限同定する。

(2) $L \notin \mathcal{L}$ ならば、M は σ から言語族 $\mathcal{L}$ を反駁する。

言語族 $\mathcal{L}$ が完全データから反駁推論可能であるとは、完全データから言語族 $\mathcal{L}$ を反駁推論する RIIM Mが存在することである。

$S, T, F \subseteq \Sigma^*$とする。$T \subseteq S$かつ $F \subseteq S^c$となるとき、Sは集合対 $I = (T, F)$ と無矛盾であるという。ただし、言語 Sの補集合を S^cで表す。

定義 2.3 [8] L を言語、$\mathcal{L}$ を言語族とする。集合対 $I = (T, F)$ が、L の $\mathcal{L}$ での 確定有限証拠集合対 (pair of definite finite tell-tales、*pdftt*) であるとは、(1) $T, F(\subseteq \Sigma^*)$ が有限集合で、 (2) L は I と無矛盾であり、かつ (3) Iと無矛盾な言語が（$L \in \mathcal{L}$ の場合はその L を除いて）族 $\mathcal{L}$ には存在しないことをいう。

$\mathcal{L}$ を 言語族とし、Σ上の有限集合の対 $I=(T,F)$ に対して

$$econs_{\mathcal{L}}(I)=\begin{cases}1, & I \text{ と無矛盾な言語が } \mathcal{L} \text{ に存在する。}\\ 0, & o.w.\end{cases}$$

定理 2.4 [9] 言語族 $\mathcal{L}$ が完全データから反駁推論可能であるための必要十分条件は、$econs_{\mathcal{L}}$ が帰納的でかつ、任意の言語 $L\notin\mathcal{L}$ に対して、L の *pdftt* が存在することである。

この定理から、正則言語の族が反駁推論可能でないことが示されている（cf.[9]）。*pdftt* の存在は反駁推論可能であるための必要条件であるが、言語族に制限を加えるとこの条件を弱めることができる。次の概念は、正データからの帰納推論可能性の特徴づけるために Angluin [1] によって導入された。

定義 2.5 [1] L を言語、$\mathcal{L}$ を言語族とする。集合 $T\subseteq\Sigma^*$ が ($\mathcal{L}$ での) 有限証拠集合 (finite tell-tale、*ftt*) であるとは、Tが L の有限部分集合でありかつ、$T\subseteq L'\subsetneq L$ となる言語 $L'\in\mathcal{L}$ が存在しないことをいう。

定義 2.6 [5] 言語族 $\mathcal{L}$ が M-有限の厚さ (M-finite thickness) を持つとは、任意の空でない有限集合 Tに対して、(1) T の $\mathcal{L}$ での極小言語の集合が有限であり、(2) 任意の $L\in\mathcal{L}$ に対して、$T\subseteq L$ であるならば、$L'\subseteq L$ となる T の極小言語 $L'\in\mathcal{L}$ が存在することをいう。

次の定理は、完全データから反駁推論可能な M-有限の厚さを持つ言語族の特徴づけ定理である。

定理 2.7 [6] $\mathcal{L}$ を M-有限の厚さを持つ言語族とする。$\mathcal{L}$ が、完全データから反駁推論可能であるための必要十分条件は、$econs_{\mathcal{L}}$ が帰納的でかつ、任意の $L\notin\mathcal{L}$ に対して L の *ftt* が存在することである。

Σ^*の *word* に演算 $\{\cup,\cdot,^*\}$ を高々 n 回施して得えられる正則言語の族を $\mathcal{L}_\Sigma(n)$ と書く。明らかに、$\bigcup_{n=0}^{\infty}\mathcal{L}_\Sigma(n)$ は正則言語全体となる。

定理 2.8 [6] 任意の $n\in N$ に対して、$\mathcal{L}_\Sigma(n)$ は完全データから反駁推論可能である。

定義 2.9 言語族 $\mathcal{L}$ が完全データから多項式時間反駁推論可能であるとは、次の条件を満たす反駁推論機械 M が存在することである：

(1) Mは完全データから $\mathcal{L}$ を反駁推論する。

(2) Mは i 番目の例を受け取った後、i 番目の出力（仮説の出力または停止）を生成し、その時間がそれまでの入力の長さに関する多項式時間である。

3 言語の生成集合

任意の正則言語は正則表現で表され、各正則表現は有限個の *word* と補助記号 $(,),^*,\cdot,+$上の文字列で表される。言語の各 *word* の基本単位はアルファベットΣであるが正則言語の場合は特に有限個の *word* 上の言語と考えることもできる。例えば、正則表現 $abc\cdot(((bb)^*+cbb))^*$ は abc,bb,cbb と、補助記号上の文字列である。本節では言語を生成する *word* の集合について考察する。

定義 3.1 $W\subseteq\Sigma^+$ が *prefix-free* 集合であるというのは、任意の Wの要素が互いに *prefix* にならないことを言う。

定義 3.2 $W \subseteq \Sigma^+$ が *simple prefix-free* (SPF と略記) であるとは、任意の $u_1, u_2 \in W$ $(u_1 \neq u_2)$ に対して $head(u_1) \neq head(u_2)$ であることをいう。ただし、$w \in \Sigma^+$の先頭の文字を $head(w)$ とする。

定義 3.3 SPF 集合 W, W'に対して、次の順序関係を導入する。

$$W \preceq W' \iff W \subseteq W'^+$$

定義 3.4 SPF 集合 Wが、言語 L の (SPF) 生成集合であるとは $L \subseteq W^*$ かつ、任意の $W' \subsetneq W$ に対して、$L \not\subseteq W'^*$ であることをいう。生成集合の要素を生成元と呼ぶ。

言語 L のすべての生成集合の集まりを $\mathcal{W}^L$とする。明らかに、$(\mathcal{W}^L, \preceq)$ は半順序集合である。$\mathcal{W}^L$に最小元 (最大元) が存在するとき、その生成集合を W^L_{inf} (W^L_{sup}) とかく。L はΣ上の言語であるから、集合 $\Sigma_L = \{u \in \Sigma \mid u$ は L に含まれる ある *word* に現れる $\}$ は 明らかに L の生成集合でありかつ、任意の $W \in \mathcal{W}^L$ に対して $W \preceq \Sigma_L$ である。すなわち、Σ_L は $\mathcal{W}^L$ の最大元であるから、$\Sigma_L = W^L_{sup}$ となる。

補助定理 3.5 L を言語とする。任意の $W, W' \in \mathcal{W}^L$ に対して、W と W'の上限および下限がそれぞれただ一つ存在する。

定理 3.6 任意の言語 L に対して、$(\mathcal{W}^L, \preceq)$ は有限束である。

系 3.7 任意の言語 L に対して、$W^L_{inf} = W^T_{inf}$ となる有限集合 $T \subseteq L$ が存在する。

系 3.8 L, L'を任意の言語とする。 $L \subseteq L'$ ならば、$W^L_{inf} \preceq W^{L'}_{inf}$ である。

生成集合の族が束をなしていることが示せた。では、実際にどのようにして生成集合を構成するのだろうか. 特に、下限となる生成集合はどう構成されるのだろうか。言語 L が有限であるときの構成方法は Tanida & Yokomori [7] による次の手続き UPDATE を用いて示される。

$\Sigma = \{a_1, \cdots, a_m\}$, $L = \{w_1, \cdots, w_n\}$ とする。任意の SPF 集合 Wは m 個の成分をもつベクトル $\mathcal{T} = (u_1, u_2, \cdots, u_m)$ で一意に表される。ただし、 $u_j \in W, head(u_j) = a_j$ であるかまたは $u_j = \lambda$ $(j = 1, \cdots, m)$ で $W = \{u_j \mid 1 \leq j \leq m, u_j \neq \lambda\}$。次の手続きは入力 wに対して、現在の $\mathcal{T}$を更新し、wを生成するようベクトルを出力する。

$u, v, x \in \Sigma^*$に対して

$$\text{common-prefix}(xu, xv) = x, \quad head(u) \neq head(v)$$

$$\text{remove-prefix}(v, u) = \begin{cases} x, & u = v \cdot x \text{ のとき} \\ \lambda, & o.w. \end{cases}$$

Procedure UPDATE$(\mathcal{T}, x)$ $\qquad\qquad \mathcal{T} = (u_1, u_2, \cdots, u_m)$

begin
  if $x \neq \lambda$ **then begin** Let $a_j := head(x)$;
    if $u_j = \lambda$ **then** $u_j := x$
    else begin
      $\alpha :=$ **common-prefix**(u_j, x); $\beta :=$ **remove-prefix**(u_j, α);
      $z :=$ **remove-prefix**(x, β); $u_j := \alpha$;
      call UPDATE$(\mathcal{T}, \beta)$; **call UPDATE**$(\mathcal{T}, z)$
        end
    end
end

この UPDATE は多項式時間で計算可能であることが示されている（ cf.[7] ）。初期ベクトル $\mathcal{T}=(\lambda,\cdots,\lambda)$ に L に含まれる *word* の列 $w_1,w_2,\cdots,w_n$ を次々に入力して、最終的に出力されるベクトル $\mathcal{T}$ を $\mathcal{T}(w_1,w_2,\cdots,w_n)$ とかく。

定理 3.9 $L=\{w_1,\cdots,w_n\},\mathcal{T}(w_1,w_2,\cdots,w_n)=(u_1,u_2,\cdots,u_m)$ とする。このとき、

$$\{u_j \mid 1\leq j\leq n, u_j\neq\lambda\}=W_{inf}^{L}.$$

系 3.10 入力列 $w_1,w_2,\cdots,w_n$ に対する *UPDATE* の出力は入力列の順序によらない。

以下、$L=\{w_1,\cdots,w_n\}$ が与えられたとき、ベクトル $\mathcal{T}(w_1,\cdots,w_n)$ に対応する生成集合（λ を除いた集合）を W_L とかく。

4 SR 言語とその特徴付け

この節では、K.Sato & M.Sato によって導入された *simple regular* オートマトンを定義し、*simple regular* 言語とはどのような言語であるかについて論ずる。

定義 4.1 *word* 上のオートマトン $M=(Q,W,\delta,p_0,F)$ が *simple regular automaton*(以下、SRA と略記）であるとは、次の条件を満たすことである:
(1) Q は状態の有限集合、(2) W は *SPF* 集合、(3) $F\subseteq Q$ は最終状態の集合、(4) δ は $Q\times W$ から Q の部分関数で、任意の $u\in W$ に対して $\delta(p,u)=q$ $(p,q\in Q)$ を満たす状態 q がただ 1 つとなっているものである。この q を u に対応する状態と呼び、p_u 或いは q_u とも表す。

SRA M によって受理される言語 L を *simple regular*(SR) 言語と呼び、SR 言語の全体を $\mathcal{L}(SR)$ とかく。

$p\in Q$, $u,v\in W$ に対して遷移関数 δ の定義域を通常の方法で $Q\times W$ から $Q\times W^*$ へ拡張する。

定義 4.2 *SRA* $M=(Q,W,\delta,p_0,F)$ が *reduced* であるとは、(1) p_0 への遷移はない、(2) 初期状態、最終状態以外の状態 $p\in Q$ に対して、p からの遷移は 2 つ以上存在する、(3) 任意の $q\in Q$ に対して、$\delta(p_0,v_1)=q,\delta(q,v_2)\in F$ となる $v_1,v_2\in W^*$ が存在することである。

M が reduced であるとき、明らかに W は $L(M)$ の生成集合となっている。

定理 4.3 [4] 任意の SRA M に対して $L(M)=L(M')$ となる reduced SRA M' が存在する。また、それを構成するためのアルゴリズムがある。

言語 L と $w\in\Sigma^*$ に対して次の集合を定義する。

$$T_L(w)=\{\ v\in\Sigma^* \mid w\in\Sigma^*, wv\in L\ \}.$$

定理 4.4 (SR 言語の特徴付け定理) L が SR 言語であるための必要十分条件は次の条件を満たす L の生成集合 W が存在することである:
任意の $v_1,v_2\in W^*, a\in\Sigma, w_1,w_2\in\Sigma^*$ に対して

$$v_1aw_1,v_2aw_2\in L\ \Rightarrow\ T_L(v_1a)=T_L(v_2a)$$

proof. $L \in \mathcal{L}(SR)$ とすると、$L(M) = L$ を満たす SRA $M = (Q, W, \delta, p_0, F)$ が存在する。$v_1aw_1, v_2aw_2 \in L$ $(v_1, v_2 \in W^*, a \in \Sigma, w_1, w_2 \in \Sigma^*)$ と仮定する。$v_1, v_2 \in \Sigma^+$ より、$aw_1 = uw_1', aw_2 = uw_2'$ で $head(u) = a$ となる $u \in W, w_1', w_2' \in W^+$ が存在する。ここで、u に対応する状態を $p_u \in Q$ とすると、M は SRA なので、$\delta(p_0, v_iu) = \delta(\delta(p_0, v_i), u) = p_u$ $(i = 1, 2)$. これは、$T_L(v_1u) = T_L(v_2u)$ であることを意味する。一方、$u = au'$ $(u' \in \Sigma^*)$ と表されるので $\{u'\} \cdot T_L(v_1a) = T_L(v_1u) = T_L(v_2u) = \{u'\} \cdot T_L(v_2a)$。したがって、$T_L(v_1a) = T_L(v_2a)$.

逆を示す。条件を満たす L の生成集合 W が存在すると仮定する。
先ず、W が SPF 集合で、しかも $L \subseteq W^*$ であるので、定理に与えられた条件は以下のように言い替えることが出来る：任意の $v_1, v_2 \in W^*, u \in W, w_1, w_2 \in W^*$ に対して

$$v_1uw_1, v_2uw_2 \in L \;\Rightarrow\; T_L(v_1u) = T_L(v_2u)$$

これの条件から、任意の $v_1, v_2 \in W^*, u \in W$ に対して、$T_L(v_1u), T_L(v_2u) \neq \phi$ ならば $T_L(v_1u) = T_L(v_2u)$ であることがいえる。従って、集合 $\{T_L(w) \mid w \in \Sigma^*, T_L(w) \neq \phi\}$ は高々 $(m+1)$ である。

ここで、言語 L を受理する SRA $M = (Q, W, \delta, p_0, F)$ を次のように構成する:

(1) $Q = \{p_0\} \cup \{\, p_u \mid u \in W\}$
(2) $F = \{p_u \mid u \in W, \exists v \in W^*,\ s.t.\ \lambda \in T_L(vu)\}$
(3) δ : $u \in W'$ に対して、$T_L(u) \neq \phi$ ならば、$\delta(p_0, u) = p_u$ とする。
$u, u' \in W$ に対して、$T_L(vuu') \neq \phi$ となる $v \in W^*$ が存在するならば、$\delta(p_u, u') = p_{u'}$ とする。

明らかに M は SRA であり、$L(M) = L$ を満たすことも容易に示される。 ∎

Angluin [2] は k-reversible 言語と呼ばれる正則言語を定義し、正データから多項式時間推論可能であることを示した。k-reversible 言語 L は、$u_1vw, u_2vw \in L, |v| = k$ ならば、$T_L(u_1v) = T_L(u_2v)$ という性質で特徴づけられる (cf. [2])。R_k をすべての k-reversible 言語とする。

定理 4.5 任意の $k \geq 0$ に対して、$L \notin R_k$ となる $L \in \mathcal{L}(SR)$ が存在する。

定理 4.6 (SR 言語と閉包性) $\mathcal{L}(SR)$ は 共通部分, *, $^+$ 演算の下で閉じているが、和集合、補集合および連接演算の下では閉じていない。

5 SR 言語の多項式時間反駁推論

この節では、SR 言語が完全データから反駁推論可能であることを示し、最後に多項式時間で反駁推論するアルゴリズムを与える。

有限集合 $T = \{\, w_1, \cdots, w_n \,\}$ を受理する SRA $M_T = (Q_T, W_T, \delta_T, p_0, F_T)$ を次のプログラムにより構成する。ただし、$\mathcal{T} = \mathcal{T}(w_1, \cdots, w_n) = (u_1, u_2, \cdots, u_m)$ に対応する生成集合を W_T, δ_T を遷移関数 δ_T の集合とする。

Procedure CONSTRUCT$(\mathcal{T}, T)$
begin
  $Q_T := \{\, p_0 \,\}$; $F_T := \phi$; $\delta_T := \phi$;
  for $i = 1$ **to** n
  begin
    if $w_i = \lambda$ **then** $F_T := F_T \cup \{\, p_0 \,\}$
    else begin $w_i = u_{k_1} \cdots u_{k_l}$; $Q_T := Q_T \cup \{\, p_{u_{k_j}} \mid j = 1, \cdots, l\}$; $\quad (u_{k_j} \in W_T, j = 1, \cdots, l)$
      for $j = 1$ **to** l
        begin
          $\delta_T := \delta_T \cup \{\, \delta_T(p_{u_{k_{j-1}}}, u_j) = p_{u_{k_j}} \,\}$;

```
            F_T := F_T ∪ { p_{u_{k_l}} }
          end
        end
    end
end
```

補助定理 5.1 CONSTRUCT は入力列 $w_1, w_2, \cdots, w_n$ の順序によらない。

有限集合 T に対する CONSTRUCT の出力を M_T とかく。

補助定理 5.2 任意の有限集合 T に対して、Tに対する CONSTRUCT の出力 M_T は $\mathcal{L}(SR)$ でのその最小の SR 言語を受理する SRA である。

定理 5.3 $\mathcal{L}(SR)$ は M-有限の厚さを持つ。

定理 5.4 $\mathcal{L}(SR)$ は完全データより反駁推論可能である。

proof. 定理 5.3 より、$\mathcal{L}(SR)$ は M-有限の厚さを持つ。従って、定理 2.7 より次の (1) および (2) を示せばよい：

(1) 任意の $L \notin \mathcal{L}(SR)$ の ftt が存在する。

(2) $econs_{\mathcal{L}(SR)}$ は帰納的である。

(1): $L \notin \mathcal{L}(SR)$ とする。L が有限のときは、L 自身がその ftt である。L が無限のとき、$w_1, w_2, \cdots$ を L の枚挙とし、$T_n = \{w_1, \cdots, w_n\}$ とする。M_{T_n} を、単に M_n とかくことにする。補助定理 5.2 より、任意の $n \in N$ に対して、$T_n \subseteq L(M_n)$ でかつ $L(M_n) \subseteq L(M_{n+1})$ である。系 3.8 より、$W_{inf}^{T_{n'}} = W_{inf}^{L}$ となる $n' \in N$ が存在する。任意の $n \geq n'$ に対して、$W_{inf}^{T_n} = W_{inf}^{T_{n'}}$ であるから、T_n に対する M_n の状態集合は $M_{n'}$ の状態集合と同じであり、M_n の遷移は $M_{n'}$ の遷移と同じであるか、または高々有限個の遷移がつけ加わっただけである。状態集合が固定されているのでそのような SRA は有限個しか存在しない。従って、$L(M_n) \subsetneq L(M_{n+1})$ となる n は有限個しか存在しない。よって 任意の $n \geq n_0$ に対して $L(M_n) = L(M_{n_0})$ となる $n_0 (\geq n')$ が存在する。 任意の $n \in N$ に対して $T_n \subseteq L(M_{n_0})$ であるから、$L \subseteq L(M_{n_0})$。一方、補助定理 5.2 から、$L(M_{T_0})$ は T_{n_0} を含む最小の SR 言語である。従って、T_{n_0} は L の ftt である。

(2): $I = (T, F)$ を任意の有限集合対とする。補助定理 5.2 より、$L(M_T)$ は T を含む最小の SR 言語であるから次の等価性がいえる：

$$econs_{\mathcal{L}(SR)}(I) = 1 \quad \Longleftrightarrow \quad L(M_T) \cap F = \phi$$

F は有限集合であるから、右辺は決定可能である。よって $econs_{\mathcal{L}(SR)}$ は帰納的である。 ∎

次のプログラムは目標言語 L の完全提示より L を反駁推論するプログラムである。

Refutable Identification Algorithm RIA

```
begin
  T := φ; F := φ; Q_T := {p_0}; W_T := φ; δ_T := φ; F_T := φ;
  T := (λ, ···, λ);                          %初期設定
  repeat
    let M_T = (Q_T, W_T, δ_T, p_0, F_T) be the current SRA;
    read the next example (w, t);            %例の入力
    if t = 0 then begin                      %負の例であったときの処理
      F := F ∪ {w};
      if w ∈ L(M_T) then stop                %負の例を受理すれば 反駁
      else output M_T                        % o.w. そのまま 出力
    end
```

```
    else begin                          %正の例であったときの処理
      if w ∈ L(M_T) then output M_T     %正の例を受理すればそのまま出力
      else begin                        % Tを含む最小の M_T を構成する
        T := T ∪ {w};
        call UPDATE(T, w);
        call CONSTRUCT(T, T);
        if F ⊈ L(M_T)^c then stop       % M_T が負の例を受理すれば 反駁
        else output M_T                 % o.w. 更新された M_T を出力
      end
    end
  until Q_T = φ
end
```

実際このプログラムは K.Sato & M.Sato [4] による SR 言語を正データから多項式時間で極限同定する推論アルゴリズム IA に $F \subseteq L^C$ であるかどうかの check 機能を加えただけのものである。従って、$L \in \mathcal{L}(SR)$ であるときは、完全提示の正の例から、L を極限同定する。$L \notin \mathcal{L}(SR)$ であるときは、完全提示の有限個の正の例 T から構成される SRA の受理する言語は T を含む $\mathcal{L}(SR)$ での最小言語であるので、いつかはこの SRA で受理されない負の例 F が現れ、その時点で反駁され停止することになる。i 番目の例 (w_i, t_i) が提示されたとき、$t_i = 1$ ならばその更新に要する時間は、$O(Nm)$ $(N = \sum_{j=1}^{i} |w|, m = |\Sigma|)$ である (cf.[4])。ここでは、さらに $F \subseteq L^c$ か否かを調べるが $F = \{w_j | j \leq i, t_i = 0\}$ であるので、その計算時間は $Nm + |F| \leq Nm + N \leq 2Nm$ より、$O(Nm)$ である。$t_i = 0$ の場合も同様に、$F \subseteq L^c$ を調べるだけであるので RIA における計算時間は $O(Nm)$ である。従って次の定理がいえる。

定理 5.5 *RIA* は *SR* 言語族 $\mathcal{L}(SR)$ を完全データから多項式時間で反駁推論する。

参考文献

[1] D. Angluin. "*Inductive Inference of Formal Languages form Positive Data*", Information and Contoral., vol. 45. 117-135, 1980.

[2] D. Angluin. "*Inference of Reversible Language*", Journal of the Association for Computing Machinary., vol. 29, No.3, pp. 741-765, July 1982.

[3] E. M. Gold. "*Language Identification in the Limit*", Information and Control., vol. 10, pp. 447-474, 1967.

[4] K. Sato & M. Sato. "正データからの *Simple Regular* 言語の多項式時間帰納推論", 数理解析研究所講究録「アルゴリズムと計算量理論」1995 年.

[5] M. Sato & T. Moriyama. "*Inductive Inference of Length-bounded EFS's form Positive Data*", DMSIS Research Report., April 11, 1994.

[6] M. Sato. "*Inductive Inference of Formal Language*", to appear in Bull. Inf. Cybern., 1995.

[7] N. Tanida & T. Yokomori. "*Polynomial-time Identification of Very Regular Language in the Limit*", Proc. 2nd Workshop on Algorithmic Learning Theory., pp. 61-72, 1991.

[8] Y. Mukouchi. "*Characterization of Finite Identification*", Proc. Workshop on Analogical and Inductive Inference., pp. 260-267, 1992.

[9] Y. Mukouchi & S. Arikawa. "*Towards a Mathematical Theory of Machine Discovery from Facts*", Theoretical Computer Science, vol.137, pp.53-84, 1995.